Chapter 13
MANAGERS

In a world where teams self-organize, is there a place for managers? Absolutely. Even though the Scrum framework doesn't specifically mention the manager role, managers still play an important part in an agile organization. After all, plenty of non-Scrum roles exist within organizations that are nonetheless crucial to the company's operations. (Accountant isn't a Scrum role, but I haven't met a Scrum team member yet who doesn't want to get paid!)

In this chapter I discuss the responsibilities of functional-area managers (also called resource managers), such as development managers, QA managers, and art directors, within a Scrum organization. I conclude by discussing the project manager role within a Scrum organization.

This chapter is more immediately relevant to organizations that have functional-area managers and project managers. If your organization is small and relatively light on managers, you can skip this chapter. However, you will probably still find it valuable to read as it will provide insight that will be necessary later on as your organization grows.

Overview

According to a 2011 industry agile survey, the number-one impediment to adopting Scrum is the feeling of a loss of management control (see Figure 13.1, from VersionOne 2011).

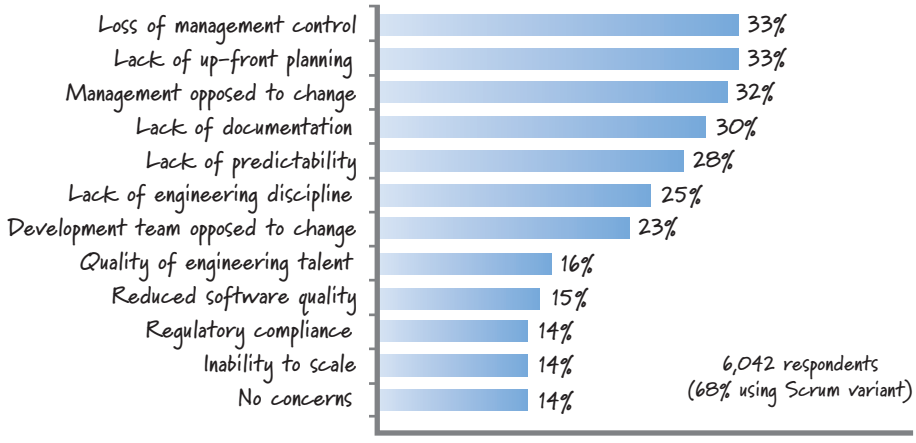

FIGURE 13.1 Greatest concerns about adopting agile

Fear that the manager role will become less relevant is unwarranted. Within a Scrum organization, the managers continue to have important responsibilities (see Figure 13.2).

In particular, functional managers in a Scrum organization are responsible for fashioning teams, nurturing teams, aligning and adapting the environment, and managing the value-creation flow.

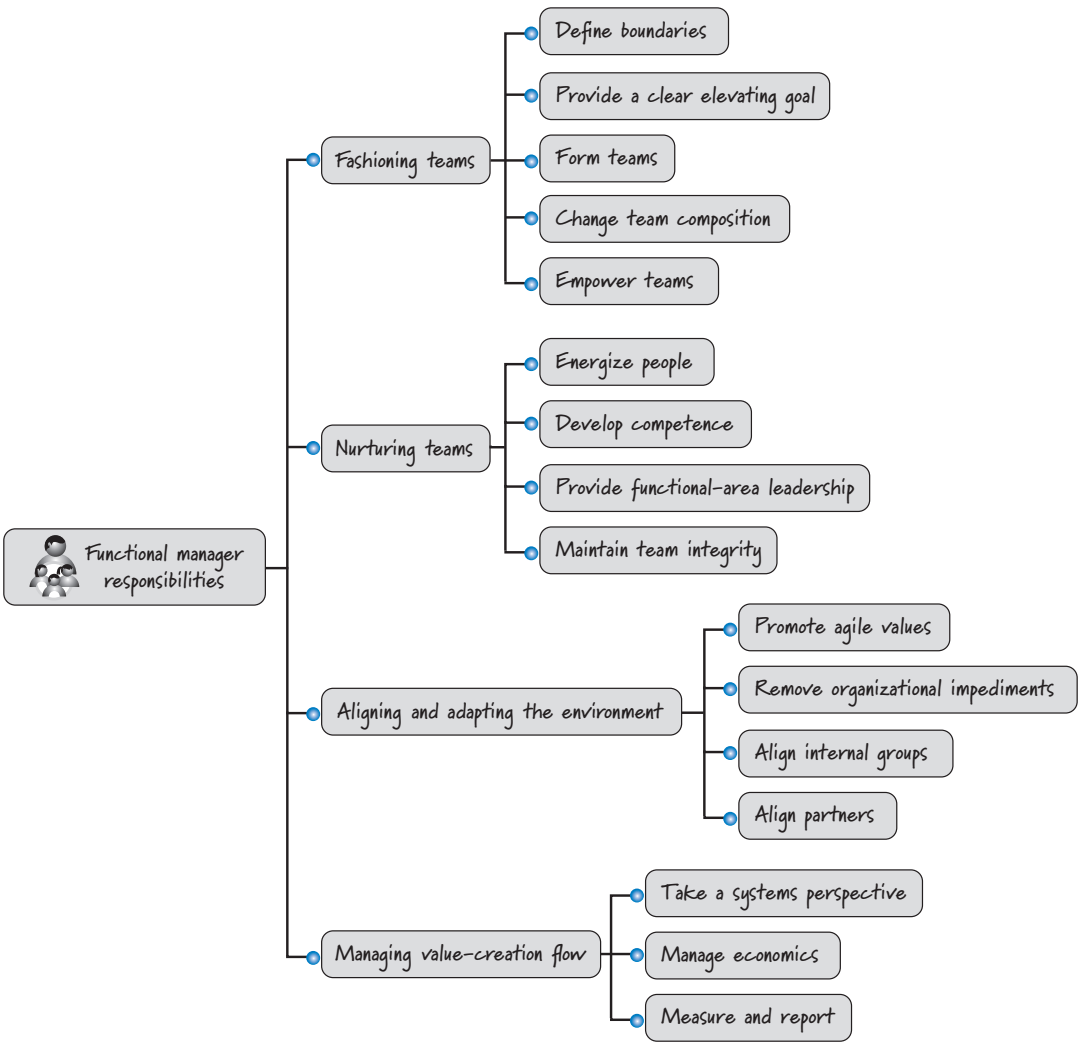

FIGURE 13.2 Functional manager responsibilities in a Scrum organization

Fashioning Teams

Managers fashion teams, the process of which includes defining boundaries, providing a clear elevating goal, forming teams, changing team composition, and empowering teams.

Define Boundaries

In Chapter 11 I described how a self-organizing team manages its response to the environment in which it is placed. The environment, however, is under the influence of managers (see Figure 13.3).

It is rare that a self-organizing team gets to decide what products or projects it will pursue. For example, if the organization builds accounting software, the team can't just decide it would like to build traffic-light control software. Managers almost always make these decisions—managers get to define the sandboxes or boundaries within which a team is permitted to self-organize.

As an example, if the teams are creating sand castles, management is deciding how many sand castles to create (how many sandboxes) and the boundaries of each sandbox, in which a specific team may self-organize and create its sand castle. Or, to use a more IT-specific example, managers in an organization that builds accounting software can decide which accounting applications to build and can set boundaries by deciding if the development teams will hand off to deployment teams for later deployment or if the development teams must deploy as part of each sprint.

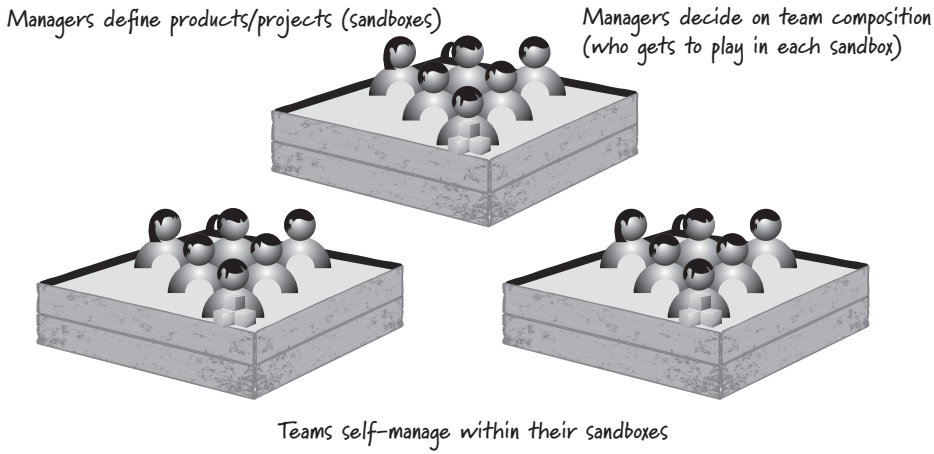

FIGURE 13.3 Managers define the boundaries.

Provide a Clear Elevating Goal

Managers also provide a clear elevating goal to each team. This goal gives purpose and direction to the team. Following the sandbox analogy, managers can decide that they want a sand castle that will win best of show at this weekend's sand castle competition, and the product owner working on the Scrum team might then further define the goal to be "Create a medieval castle, complete with turrets and a surrounding moat."

Form Teams

Teams do not typically form themselves (team members do not self-select who will be on the team). Managers compose teams. Returning to our sandbox analogy, this means that managers almost always decide who gets to play in each sandbox, not the individual team members themselves. Certainly team members can and should provide input into the team formation process—for example, by requesting to be on a particular team or by interviewing new candidates for an existing team. However, in most organizations managers make the final decision to ensure that team formation properly balances business needs and constraints.

In a Scrum environment, functional managers representing different disciplines or communities of practice work with one another to select members of cross-functional Scrum teams (see Figure 13.4).

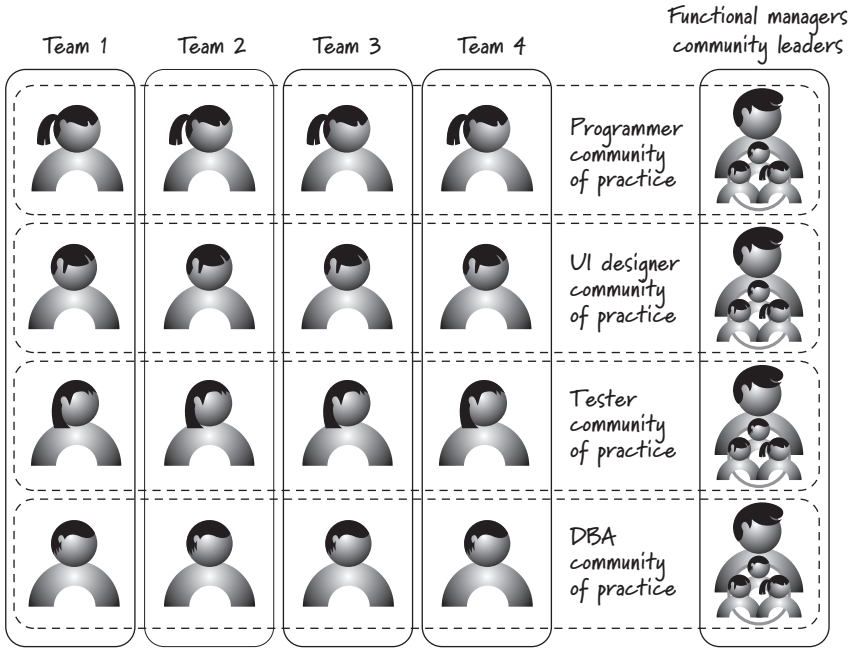

FIGURE 13.4 Functional managers collectively create Scrum teams.

In this figure each horizontal band represents a functional area or community of practice composed of people with similar specialty skills (for example, a community of developers, UI designers, testers, or DBAs). Each functional area has a functional manager.

The functional managers are collectively responsible for selecting the proper people from each functional area to form Scrum teams, which are shown vertically in the figure. Managers strive to form teams that are cross-functionally diverse and sufficient, where the team members have a good complement of T-shaped skills (see Chapter 11).

Change Team Composition

Managers also have the obligation to change a team's composition if they believe that doing so will improve the overall health and performance of the team and the organization as a whole.

Let's say, for instance, that Fred is a low-performing person on the development team. Fred also has a bad attitude and is negatively affecting the team's ability to perform. How should Fred's situation be handled?

First, I would expect Fred's teammates to discuss the situation with him with the goal of trying to help him and the team. If they are unsuccessful, the ScrumMaster, as Scrum team coach, would work with Fred to help him be a more effective team member. If coaching doesn't work, Fred's situation would most likely escalate out of the Scrum team to his resource manager (the person to whom Fred reports within the organization), because the ScrumMaster does not have hiring and firing authority.

At this point Fred's resource manager (perhaps in conjunction with someone from human resources) would handle his performance issues in a humane and appropriate manner. The resource manager would certainly want to consult the ScrumMaster and development team members to deepen his understanding of the situation. At that point the resource manager might decide to immediately remove Fred from the Scrum team and assign him to another team where he might be a better fit. Alternatively, he could put Fred on a performance improvement plan (either on his current team or on a new team), and if Fred doesn't improve per the plan, he might be let go.

While managers have "firing" authority and team members and ScrumMasters do not, team members certainly are involved in the process of ensuring that the team is well fashioned.

Managers may also have to alter team composition when doing so better optimizes the organization's ability to deliver across its portfolio of products. For example, even though we prefer our teams to be long-lived, it may be necessary from time to time to move a person who has a special set of skills off of one team and onto another team that has an immediate, high-value need for those skills. Managers need to make such changes with care because both teams will be affected by the change in team member composition.

Empower Teams

For teams to self-organize they must be empowered, which requires authorization and trust from managers. One principal way of empowering teams is for managers to delegate responsibilities to them with the primary goal of allowing self-organizing teams to manage themselves better. That being said, teams don't get to make all of the "management" decisions (as we discussed earlier, Fred's team members cannot fire him for being a poor performer). However, teams can be empowered to take on typical management activities.

For each activity type or specific decision the manager might consider delegating, he picks the proper level of authority for empowering the team. Appelo defines seven levels of authority as shown in Table 13.1, each with an example (Appelo 2011).

These levels range from one extreme, *tell*, where the manager makes the decision and informs the team, to the other extreme, *delegate*, where the team has full authority to make the decision.

When managers delegate tasks, they must trust that the teams will carry out their responsibilities as expected. And the teams must trust that their managers will not take actions that are in contradiction to delegated authority. For example, managers should not delegate authority for a decision to the team and then go off and make the decision.

TABLE 13.1 Appelo's Seven Levels of Authority, with Examples

Level	Name	Description	Example
1	Tell	Manager makes the decision and tells the team	Relocate to a new office building
2	Sell	Manager convinces the team about the decision	Decision to use Scrum
3	Consult	Manager gets input from the team before making the decision	Select new team members
4	Agree	Manager and team make the decision together	Choose logo for business unit
5	Advise	Manager advises to influence the decision made by the team	Select architecture or component
6	Inquire	Manager inquires after the team has made the decision	Sprint length
7	Delegate	Manager fully delegates the decision to the team	Coding guidelines

Managers should also help the team members trust each other. They can do this by defining the proper boundary for the environment in which the team operates, which will help intra-team trust to form by setting limits on how far trust must be extended. Managers should also help team members understand the importance on a self-organizing team of meeting personal commitments, because there is no manager within the team to pressure people to get work done. And managers should reinforce a Musketeer attitude among team members, so that they can trust that everyone truly is committed to working together to meet team goals.

Nurturing Teams

Once Scrum teams have been fashioned, managers then must nurture them. By nurture I don't mean that managers should *manage* the teams. Instead, managers should energize people, focus on competence development, provide functional-area leadership, and maintain team integrity.

Energize People

Providing a clear elevating goal creates a foundation upon which to energize team members. By energize I mean that managers should constantly seek ways to motivate people to intrinsically want to do great work. We all want to work in a fun, creative, value-delivering environment, and managers are responsible for nurturing that environment. Through proper management, managers can positively influence the intrinsic motivation and energy of team members.

Conversely, managers can take actions that have the opposite affect—that sap energy from the environment and lead to demotivated people. For example, historically, functional managers are accustomed to assigning task-level work to people in their area. Doing so in a Scrum environment would deenergize people by undermining the foundation of team self-organization and compromising the ability of the team to deliver value.

Develop Competence

Within Scrum organizations, each team member still reports to a functional or resource manager who is typically not the ScrumMaster or the product owner. And, just as in non-Scrum environments, these managers take an active role in coaching and assisting their direct reports with their career goals by promoting opportunities for competency development and providing frequent, actionable feedback on performance.

Managers need to foster an environment where people are constantly learning and adding to their skill sets. They need to make it clear that learning is not only encouraged but is in fact a priority at the individual, team, and organizational levels.

Actions such as providing team members with time for training or attending conferences will speak louder than words. Within this supportive learning environment, managers coach team members to advance their domain knowledge, technical knowledge, thinking skills, and so on.

Managers must also provide frequent feedback to teams and individuals. In many non-Scrum organizations, performance feedback comes in the form of the annual performance review. In organizations using Scrum that choose to perform these reviews, the functional managers would be expected to continue the process. However, organizations that have internalized core Scrum values and principles soon realize that providing performance feedback to individuals once (or twice) a year is out of sync with the cadence of Scrum teams that are performing and learning in short-duration sprints. These annual performance reviews can also foster low-trust competition in the team rather than self-organization with a Musketeer attitude. Individual performance measures can also interfere with superior team performance by driving independent behavior—people optimize how they are personally being measured at the expense of the team. Successful Scrum organizations soon begin to question the value of even doing these annual performance reviews, realizing they may indeed cause more harm than good.

That doesn't mean individual performance isn't assessed in Scrum organizations. Managers should just align the frequency of their feedback to individuals to better match that of the learning loops of the team of which their direct reports are members. One such approach would be for managers to provide feedback every sprint. Individual feedback should also be well positioned in the context of how individual performance is supportive (or not) of team performance.

Provide Functional-Area Leadership

As in non-Scrum organizations, functional managers in Scrum organizations continue to provide leadership specific to their functional area.

Functional managers usually have good working knowledge of their functional area and can provide thought leadership within the area. This type of leadership does not involve a functional manager assigning tasks to his direct reports or telling them how to do their jobs. Such actions would be debilitating to a self-organizing team. This type of leadership does, however, support an important need for consistency, coherence, and coaching within the functional area.

For example, in game development companies, the artists report to the art director, who himself is a highly skilled artist. The art director provides art leadership to the artists by helping to set art standards for the game and then reviewing the work of individual artists to ensure holistic consistency. We don't want to have an artist on one Scrum team doing gothic art and an artist on another team doing cartoon art. The art director provides overall leadership within the area to help better ensure high-value, coherent results.

Functional managers also provide leadership by establishing area-relevant standards and by encouraging initiatives specific to their functional area. For example, let's say the QA director wants to select new test automation tools that can be used across multiple development efforts. To accomplish this, the QA director may ask the QA-centric people who report to him, but who are all members of different Scrum teams (as illustrated in Figure 13.4), to collaborate on the selection.

Maintain Team Integrity

In Chapter 11 I stated that the currency of agile is the team. Because the team replaces the individual as the unit of capacity, managers should work proactively to maintain team integrity. That means not pulling people off of teams mid-sprint to work on pet projects or unnecessarily assigning people to work on multiple teams.

Because the economics of long-lived teams are compelling, managers should also try to keep teams together as long as the economics justify doing so. At the end of a development effort, managers should first try to assign the team as a whole to the next development effort. They should do this before they impair a high-value asset by breaking it into pieces and losing the value-added cohesion of the team.

Aligning and Adapting the Environment

Getting a single team, or even the IT or development departments, to use Scrum is a good start. However, to realize the extraordinary benefits of Scrum, the entire value chain from suppliers to customers needs to embrace agile. Managers are responsible for aligning and adapting the environment (the value chain) by promoting agile values, removing organizational impediments, aligning internal groups, and aligning partners.

Promote Agile Values

Managers must embrace agile values and principles. They need to understand and truly believe them, live them, and encourage others to do the same. Far too often when I teach classes or coach Scrum teams I hear, "Yes, all of this makes sense to us, but we need to get our management to buy in or we won't be able to really do Scrum. I wish they were in the room to hear this." These teams are correct. They will eventually need management support if they are to be successful over the long term.

Once I was engaged in a lunchtime discussion with the management team of an organization that was just starting to adopt Scrum. During our discussion I remarked that managers should avoid pulling someone off an in-flight team to temporarily work on some other project because of the disruption it would create. Timidly, but with true sincerity, a manager in the room said, "OK, but I do that all the time and didn't think it was a bad thing. What are the other things I should know as a manager

in an emerging agile organization so that I can better align my behavior and the environment to promote agility?"

In response to her question I began a discussion of core agile values and principles (similar to Chapter 3) to give her and her colleagues awareness of how a manager can help reinforce agile principles instead of unknowingly working in contradiction to them. Of course, only through their day-to-day behaviors can managers truly promote agile values.

Remove Organizational Impediments

Managers also work hand in hand with ScrumMasters to remove impediments. Though the ScrumMaster is the person pushing to remove organizational impediments, many impediments, especially those that are environmental in nature, require intervention from managers to actually be removed.

Align Internal Groups

The engineering or IT group is often the first to adopt Scrum. Let's say that after a reasonable period of time the first-adopter group becomes very skilled at creating customer-valuable features each sprint. However, until those features are actually made available to customers, no real value has been delivered. What if the deployment group does not operate in an agile way, and if pushing features into production every few weeks just isn't something that group can or is willing to do? Can the organization really claim to be a high-performance Scrum organization if it can't get the value into customers' hands in a timely way?

What if there is the same sort of misalignment upstream of development? Perhaps sales and marketing are operating on a different set of principles. What if their attitude is "You guys in development can use whatever process you want to build things. You just need to be able to answer all of my up-front, very detailed questions and meet the date we already provided to the customers." Or perhaps the folks in HR are still recruiting people for old job descriptions rather than targeting people who have T-shaped skills and a desire to work in self-organizing teams.

We can't realize the full, long-term potential benefits of Scrum in such environments. Managers (including executives) have the obligation to fashion the environment in order to achieve good internal alignment among the various groups, such as governance, finance, sales, marketing, deployment, and support. Managers must see the whole and align the whole with agile principles.

Align Partners

Why stop at internal alignment? Managers must also help the organization embrace a more agile approach to supplier management and outsourcing. If the way we engage

our external partners follows a traditional arms-length, contract-heavy, negotiation-style approach, the organization will fail to achieve its full potential with Scrum.

Instead, managers should promote the use of agile principles when engaging partners. For example, the simplest form of outsourcing agreement is to lease the Scrum team of a third party. Instead of doing all of the difficult work of creating a high-performance team, managers buy access to a high-performance team that others have already created. At that point, the organization uses Scrum as described in this book, but the development team (and perhaps the ScrumMaster) is "owned" by a third party and not the organization.

To achieve this level of agile-partner alignment, managers should consider alternatives to writing fixed-priced contracts with outsourcers. Such contracts put the organization and its contractors immediately at odds with one another. (The contractor wants to deliver as little as possible to meet the contract so it can maximize its gross margins, and the organization wants to get as much as possible for the fixed price.) This is hardly an agile way of operating. Managers should change this style of engagement.

Managing Value-Creation Flow

Overall, managers in a Scrum environment are responsible for setting strategic direction and for ensuring that organizational resources are being marshaled in an economically sensible way to achieve strategic goals. Managers manage the value-creation flow by taking a systems perspective, managing economics, and measuring and reporting.

Take a Systems Perspective

To effectively manage the flow of value creation, managers must take a systems perspective. One of the larger impediments I have seen to successful Scrum adoption is when managers refuse to think systemically and instead focus only on their own areas or fiefdoms. I often hear, "Yes, but doing what you propose would require a change in the reporting structure or in key job descriptions." When people say this, what I hear is "I can't imagine that we would actually do those things, so I can't [or won't] make the change in my area to better align what we do with Scrum values and principles and the rest of the agile organization."

Such localized thinking makes it difficult to achieve any sort of sensible internal agile alignment and can lead to different parts of the organization quite literally working against the greater good of the system. Managers in a Scrum organization must be willing to take a see-the-whole perspective if they are to realize long-term, high-performance Scrum benefits.

Manage Economics

Organizations expect managers to be trusted stewards of the financial resources that are made available to them. Higher-level managers in a Scrum organization therefore still manage economics (such as profit and loss) for their areas. Functional managers or resource managers may not have direct profit responsibility but are still held accountable for how the financial resources entrusted to them are being spent.

Managers (perhaps at the executive level) are also expected to oversee economics at the higher level of the organization. This frequently occurs through their involvement in portfolio management and corporate governance. Through portfolio management, they determine which development efforts to fund, to what degree, and the order in which they should be done. And, once an effort is under way, managers review and react to the continuous stream of real-time feedback based on iterative and incremental development and, when appropriate, terminate an effort whose economics no longer justify additional expenditures (see Chapter 16).

Monitor Measures and Reports

Many measures and reports are collected and generated at the request of managers, so there is a real opportunity for managers to ensure that only those measures that add to the value-creation flow are captured and reported. This goal can be achieved by ensuring that measures and reporting align well with core Scrum values and principles.

In Chapter 3 I described several Scrum principles that can guide how managers approach measuring and reporting. The following are a few examples:

- Focus on idle work, not idle workers. To accomplish this, measure when and how often the flow of work is being impeded rather than how good you are at keeping people busy. A measure such as cycle time will expose the length of time between when work starts and when it finishes. If cycle time is increasing, you need to investigate why.
- Measure progress by working, validated assets. Does it really matter if you deliver on time and on budget if you don't deliver a product that people want? Focus on measuring the value delivered (working and validated assets), but don't lose sight of the variables (date, scope, budget, and quality) needed to deliver value.
- Organize flow for fast feedback. Align your measures to determine how quickly the learning cycle is completed (assume, build, feedback, inspect, and adapt).

This last measure is at the heart of **innovation accounting**, which is effective in any organization that creates a product or service under conditions of extreme uncertainty (Ries 2011). Innovation accounting uses actionable metrics to evaluate

how fast we are learning as a critical measure of our progress toward converging on a business-valuable result. Innovation accounting is based on three steps:

1. Create a minimum viable product (MVP) to establish actual baseline values of the actionable metrics on where the organization or product is today.
2. Using a series of incremental improvements to the product, try to improve the actionable measures from the baseline toward the ideal or desired values.
3. If actionable measures show that the product is making demonstrable progress toward the desired target, persevere on the current path; otherwise **pivot** to a new strategy and begin the process again.

I will discuss the concepts of pivot and persevere in more detail in Chapter 14, Chapter 16, and Chapter 17.

Project Managers

So far we have been discussing the role of functional manager or resource manager. What about the project manager role? Is there such a role in a Scrum organization?

Project Management Responsibilities on a Scrum Team

A common misperception is that the ScrumMaster is really just the "agile project manager" or a project manager with a different title. On the surface there are some similarities between a ScrumMaster and a project manager—for example, both do impediment removal. However, being a servant leader significantly differentiates this role from a more command-and-control-focused project manager.

To answer the question "Where is the project manager?" let's look at the core project management responsibilities as defined by the Project Management Institute (PMI 2008) and summarized in Table 13.2.

TABLE 13.2 Traditional Project Management Responsibilities

Project Management Activity	Description
Integration	Identify, define, combine, unify, and coordinate the various processes and project management activities.
Scope	Define and control what is and is not included in the project, ensuring that the project includes all of the work required.

continues

TABLE 13.2 Traditional Project Management Responsibilities (*Continued*)

Project Management Activity	Description
Time	Manage timely completion of the project by defining what to do, when to do it, and what resources are necessary.
Cost	Estimate, budget, and control costs to meet an approved budget.
Quality	Define quality requirements and/or standards, perform quality assurance, and monitor and record results of quality-focused activities.
Team (human resource)	Organize, manage, and lead the project team.
Communications	Generate, collect, distribute, store, retrieve, and dispose of project information.
Risk	Plan, identify, analyze, respond, monitor, and control project risks.
Procurement	Acquire products, services, or results needed from outside the project team.

Certainly these responsibilities remain important and need to be addressed. So, if there is no project manager, who oversees these activities?

Table 13.3 shows that these traditional project manager responsibilities are distributed among the various Scrum team roles and possibly other managers.

TABLE 13.3 Mapping of Project Management Responsibilities in a Scrum Organization

Project Management Activity	Product Owner	ScrumMaster	Development Team	Other Manager
Integration	✔			✔
Scope	Macro level		Sprint level	
Time	Macro level	Helps Scrum team use time effectively	Sprint level	
Cost	✔		Story/task estimating	

TABLE 13.3 Mapping of Project Management Responsibilities in a Scrum Organization (*Continued*)

Project Management Activity	Product Owner	ScrumMaster	Development Team	Other Manager
Quality	✔	✔	✔	✔
Team (human resource)			✔	Formation
Communications	✔	✔	✔	✔
Risk	✔	✔	✔	✔
Procurement	✔			✔

Based on Table 13.3, a person who was a project manager might assume any of the three Scrum roles, depending on that person's skills and desire. Many project managers make excellent ScrumMasters, if they can forgo any command-and-control management tendencies.

However, as you can see from Table 13.3, the product owner assumes at least as many project management responsibilities as the ScrumMaster. So, project managers can also make a transition into the role of product owner, if they have the proper domain knowledge and other skills to execute the product owner role. Or, less frequently, a project manager with a technical background might choose to become a member of the development team.

Retaining a Separate Project Manager Role

It would seem that project managers might become ScrumMasters, product owners, or team members. That is not, however, always the case. Companies that have large and complex development efforts sometimes decide to retain a separate project manager when logistics and coordination tasks are so overwhelming that the teams cannot be expected to keep up with them.

As a rule, I want the Scrum teams on a development effort to handle their own logistics and coordination. Scrum teams should not expect that someone external to the teams is responsible for coordinating things on their behalf. That expectation leads to team members thinking, "If someone else is responsible for coordinating, then we aren't."

The logistics and dependencies of smaller development efforts with only a few Scrum teams are easily handled via day-to-day inter-team coordination (using a technique like the scrum of scrums, see Chapter 12). However, what if we have a development effort with tens or hundreds of Scrum teams and hundreds or maybe a thousand developers?

Much like the one-product-one-backlog rule in Chapter 6, the teams-should-handle-their-own-coordination rule is the correct starting place. However, just as the issue of scale might cause us to relax the one-product-one-backlog rule, so might it cause us to retain one or more project or program managers to help coordinate all of the moving parts.

Before we rush down the path of retaining a coordination-specific role just because we have a lot of teams, we should step back and look at the inter-team communication channels. My experience with these situations is that rarely are the communication channels fully connected among all teams (see Figure 13.5).

More likely the teams cluster (or should cluster) together into feature areas or some equivalent, where the communication channels are more intense within a given cluster and more lightly coupled across clusters (see Figure 13.6).

In such cases, the Scrum teams can easily manage their own inter-team coordination. But who owns the cross-cluster coordination? The default answer is the teams themselves. In many cases this approach works just fine. The collaboration can be

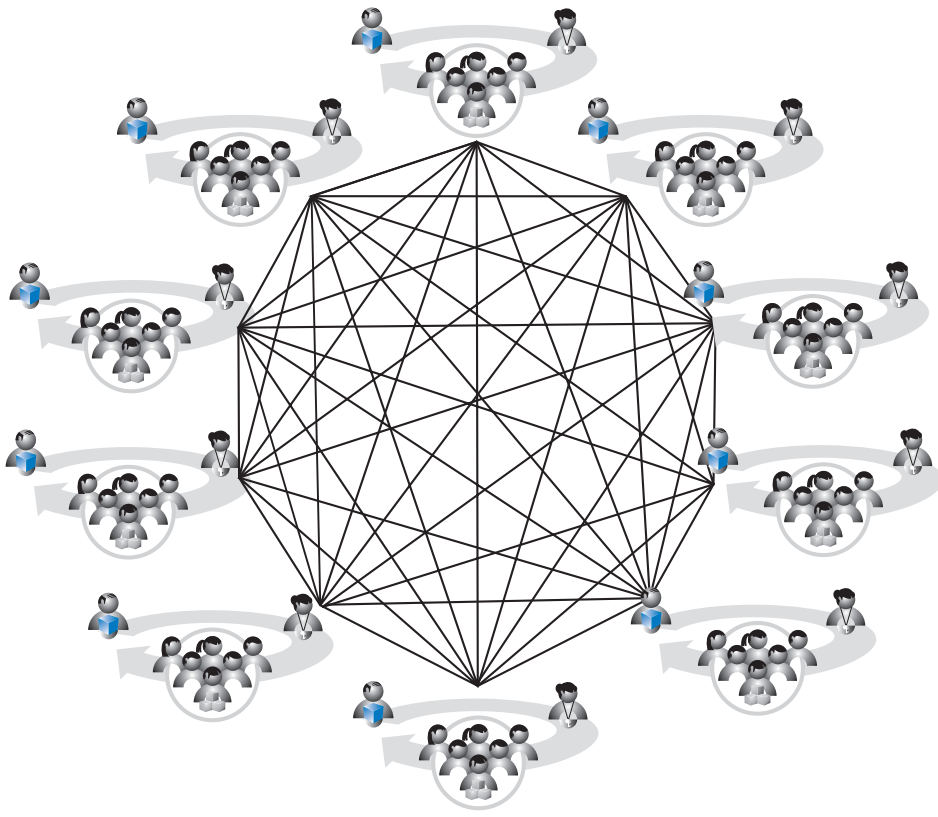

FIGURE 13.5 Teams rarely have fully connected communication channels.

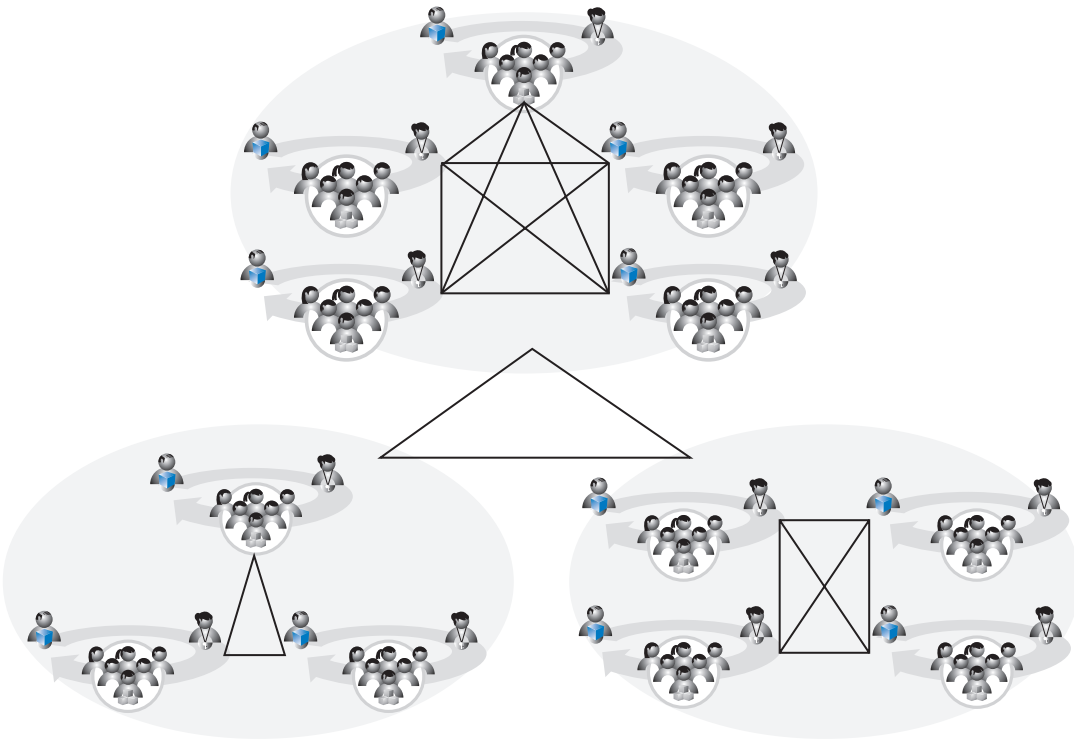

FIGURE 13.6 Teams frequently form collaboration clusters.

handled like a scrum of scrums where a representative of each cluster meets with representatives of the other clusters to discuss dependency coordination.

However, in the presence of many different clusters, even this scrum-of-scrum type of coordination among the teams might prove difficult. In such cases I have seen organizations funnel the coordination effort through a project or program manager (see Figure 13.7).

I would prefer not to have a project manager at the center of the coordination. Such an approach runs the risk of the individual Scrum teams handing off responsibility for coordination to a third party.

However, at a sufficiently large scale I do recognize that having a person or people who focus full-time on overseeing logistics and coordination can provide a level of perceived comfort that the baton won't get dropped. To clarify that the individual teams cannot delegate their inter-cluster coordination responsibilities to someone else, I prefer to think of the project manager as an assistant (like a servant leader) to the Scrum teams. In this role, the project manager is expected to have the whole-system perspective and to work diligently with each of the clusters or individual teams to ensure that everyone has the appropriate understanding of what cross-team coordination is required—but the teams still own the coordination.

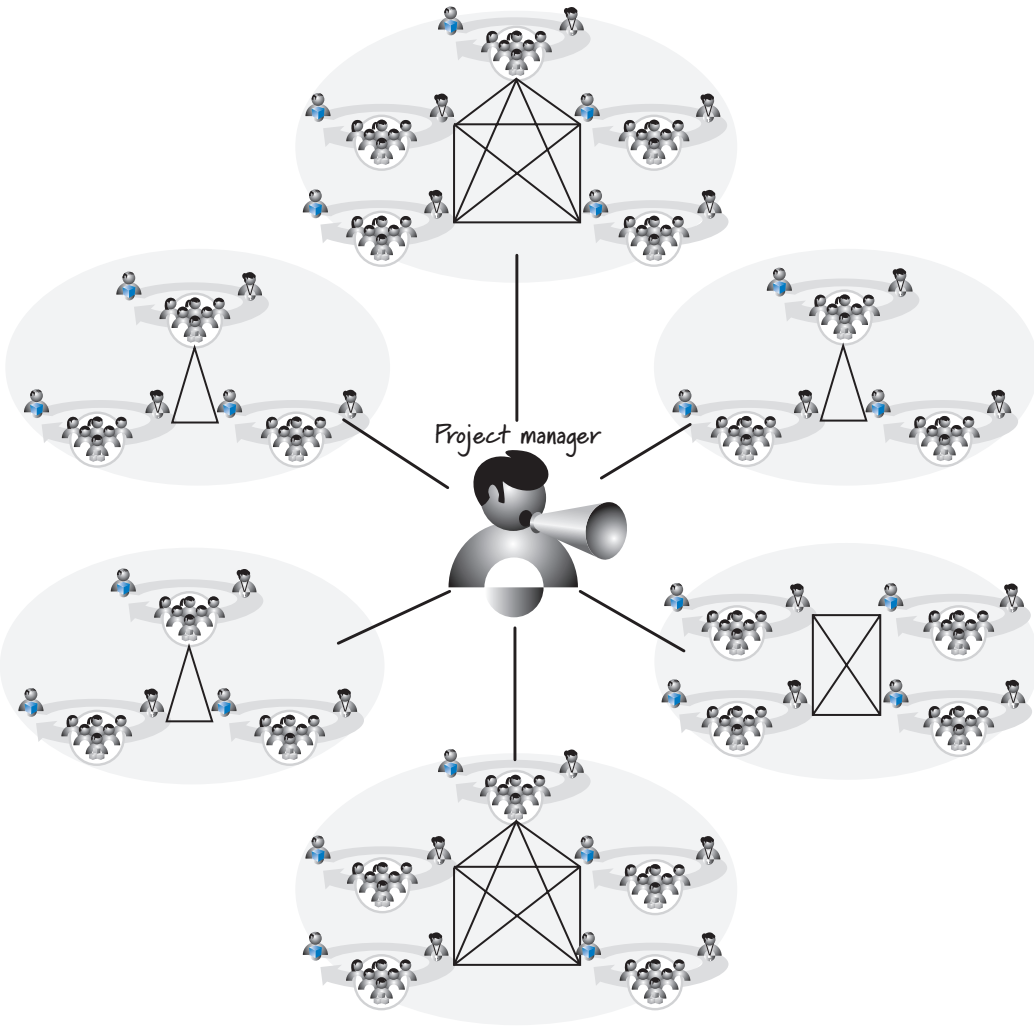

FIGURE 13.7 Funneling coordination through a project or program manager

This same use of a project manager can also be helpful on development efforts where using Scrum represents just one small part of a much greater product or services development. For example, there might be subcontractors, internal non-Scrum teams, and other internal organizations associated with delivering the product. In particular, the logistics of dealing with subcontractors or suppliers can be quite involved and time-consuming. With so many moving parts, it is helpful to have someone focused solely on the logistics (see Figure 13.8).

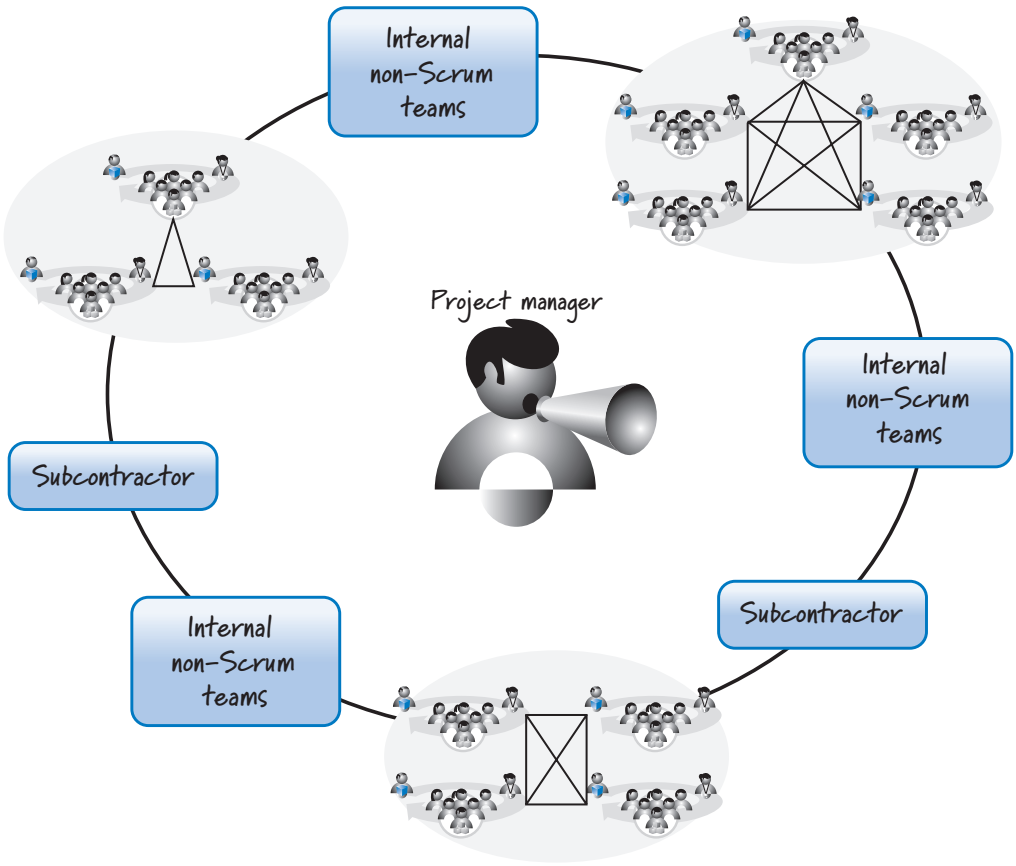

FIGURE 13.8 Project manager on complex, multiparty development

Once again, the goal isn't for the project manager to be in charge. Rather, the project manager is the person who is concerned with making sure that dependencies across the various areas are understood and communicated in a way so that the teams can most effectively coordinate their work with the other teams.

Closing

In this chapter I described the role of functional managers within a Scrum organization. I grouped managerial responsibilities into the categories of fashioning teams, nurturing teams, aligning and adapting the environment, and managing the value-creation flow.

Table 13.4 summarizes the responsibilities of functional managers in a traditional organization and those of functional managers in a Scrum organization.

TABLE 13.4 Comparison of Functional Manager in Traditional and Scrum Environments

Traditional	Scrum
Assigns people to projects	Collectively fashions great teams
Hires and fires	Same
Focuses on people development	Same
Reviews performance	Still involved but the frequency of feedback is significantly higher and feedback must be tied back to the team
Assigns tasks to team members (sometimes)	Lets team members self-organize and define and select their own tasks
Establishes cross-project standards in functional area	Same
Encourages functional-area-specific initiatives	Same
Has good working knowledge of the functional area and can lend a hand when necessary	Same
Is skilled at moving direct reports from team to team	Focuses on maintaining team integrity
Removes impediments	Same
Focuses on own functional area	Takes a see-the-whole perspective for alignment and value creation
Manages economics (P&L)	Same
Monitors measures and reports	Aligns measures and reports with agile principles to focus on value-creation flow

Although the majority of this chapter was focused on the role of functional manager, I ended with a discussion of the role of project manager. I focused on both how the traditional responsibilities of this role are shared among the three Scrum team roles and how on complex development efforts some organizations find it helpful to have one or more project managers in addition to the three Scrum roles.

This chapter concludes Part II. In the next chapter I will begin the discussion of planning by describing important Scrum planning principles.